第3章 Redis命令

Redis根据命令所操作对象的不同，可以分为三大类：对Redis进行基础性操作的命令，对Key的操作命令，对Value的操作命令。

1 Redis基本命令

首先通过redis-cli命令进入到Redis命令行客户端，然后再运行下面的命令。

心跳命令 ping

键入ping命令，会看到PONG响应，则说明该客户端与Redis的连接是正常的。该命令亦称为心跳命令。

读写键值命令

set key value会将指定key-value写入到DB。get key则会读取指定key的value值。关于更多set与get命令格式，后面会详细学习。

DB切换select

Redis默认有16个数据库。这个在Redis Desktop Manager（RDM）图形客户端中可以直观地看到。默认使用的是0号DB，可以通过select db索引来切换DB。

删除当前库中数据flushdb

flushdb命令仅仅删除的是当前数据库中的数据，不影响其它库。

删除所有库中数据命令flushall

flushall命令可以删除所有库中的所有数据。所以该命令的使用一定要慎重。

退出客户端命令

使用exit或quit命令均可退出Redis命令行客户端。

2 Key操作命令

Redis中存储的数据整体是一个Map，其key为String类型，而value则可以是String、Hash表、List、Set等类型。

keys

l 格式：KEYS pattern

l 功能：查找所有符合给定模式 pattern 的 key，pattern为正则表达式。

l 说明：KEYS的速度非常快，但在一个大的数据库中使用它可能会阻塞当前服务器的服务。所以生产环境中一般不使用该命令，而使用scan命令代替。

exists

l 格式：EXISTS key

l 功能：检查给定 key 是否存在。

l 说明：若 key 存在，返回 1 ，否则返回 0 。

del

l 格式：DEL key [key ...]

l 功能：删除给定的一个或多个 key 。不存在的 key 会被忽略。

l 说明：返回被删除 key 的数量。

rename

l 格式：RENAME key newkey

l 功能：将 key 改名为 newkey。

l 说明：当 key 和 newkey 相同，或者 key 不存在时，返回一个错误。当 newkey 已经存在时，RENAME命令将覆盖旧值。改名成功时提示 OK ，失败时候返回一个错误。

move

l 格式：MOVE key db

l 功能：将当前数据库的 key 移动到给定的数据库 db 当中。

l 说明：如果当前数据库(源数据库)和给定数据库(目标数据库)有相同名字的给定 key ，或者 key 不存在于当前数据库，那么 MOVE 没有任何效果。移动成功返回 1 ，失败则返回 0 。

type

l 格式：TYPE key

l 功能：返回 key 所储存的值的类型。

l 说明：返回值有以下六种

- none (key不存在)
- string (字符串)
- list (列表)
- set (集合)
- zset (有序集)
- hash (哈希表)

expire与pexpire

l 格式：EXPIRE key seconds

l 功能：为给定 key 设置生存时间。当 key 过期时(生存时间为 0)，它会被自动删除。expire的时间单位为秒，pexpire的时间单位为毫秒。在 Redis 中，带有生存时间的 key 被称为“易失的”(volatile)。

l 说明：生存时间设置成功返回 1。若 key 不存在时返回 0 。rename操作不会改变key的生存时间。

ttl与pttl

l 格式：TTL key

l 功能：TTL, time to live，返回给定 key 的剩余生存时间。

l 说明：其返回值存在三种可能：

- 当 key 不存在时，返回 -2 。
- 当 key 存在但没有设置剩余生存时间时，返回 -1 。
- 否则，返回 key 的剩余生存时间。ttl命令返回的时间单位为秒，而pttl命令返回的时间单位为毫秒。

persist

l 格式：PERSIST key

l 功能：去除给定 key 的生存时间，将这个 key 从“易失的”转换成“持久的”。

l 说明：当生存时间移除成功时，返回 1；若 key 不存在或 key 没有设置生存时间，则返回 0。

randomkey

l 格式：RANDOMKEY

l 功能：从当前数据库中随机返回(不删除)一个key。

l 说明：当数据库不为空时，返回一个key。当数据库为空时，返回nil。

scan

l 格式：SCAN cursor [MATCH pattern][COUNT count][TYPE type]

l 功能：用于迭代数据库中的数据库键。其各个选项的意义为：

- cursor：本次迭代开始的游标。
- pattern ： 本次迭代要匹配的key的模式。
- count ： 本次迭代要从数据集里返回多少元素，默认值为 10 。
- type：本次迭代要返回的value的类型，默认为所有类型。

SCAN命令是一个基于游标cursor的迭代器：SCAN命令每次被调用之后，都会向用户返回返回一个包含两个元素的数组， 第一个元素是用于进行下一次迭代的新游标， 而第二个元素则是一个数组， 这个数组中包含了所有被迭代的元素。用户在下次迭代时需要使用这个新游标作为SCAN命令的游标参数，以此来延续之前的迭代过程。当SCAN命令的游标参数被设置为0时，服务器将开始一次新的迭代。如果新游标返回 0 表示迭代已结束。

l 说明：使用间断的、负数、超出范围或者其他非正常的游标来执行增量式迭代不会造成服务器崩溃。

当数据量很大时，count的数量的指定可能会不起作用，Redis会自动调整每次的遍历数目。由于scan命令每次执行都只会返回少量元素，所以该命令可以用于生产环境，而不会出现像 KEYS 命令带来的服务器阻塞问题。

增量式迭代命令所使用的算法只保证在数据集的大小有界的情况下迭代才会停止，换句话说，如果被迭代数据集的大小不断地增长的话，增量式迭代命令可能永远也无法完成一次完整迭代。即当一个数据集不断地变大时，想要访问这个数据集中的所有元素就需要做越来越多的工作， 能否结束一个迭代取决于用户执行迭代的速度是否比数据集增长的速度更快。

l 相关命令：另外还有3个scan命令用于对三种类型的value进行遍历。

- hscan：属于Hash型Value操作命令集合，用于遍历当前db中指定Hash表的所有field-value对
- sscan：属于Set型Value操作命令集合，用于遍历当前db中指定set集合的所有元素
- zscan：属于ZSet型Value操作命令集合，用于遍历当前db中指定有序集合的所有元素（数值与元素值）

3 String型Value操作命令

Redis存储数据的Value可以是一个String类型数据。String类型的Value是Redis中最基本，最常见的类型。String类型的Value中可以存放任意数据，包括数值型，甚至是二进制的图片、音频、视频、序列化对象等。一个String类型的Value最大是512M大小。

set

l 格式：SET key value [EX seconds | PX milliseconds] [NX|XX]

l 功能：SET除了可以直接将key 的值设为 value外，还可以指定一些参数。

- EX seconds：为当前key设置过期时间，单位秒。等价于SETEX命令。
- PX milliseconds：为当前key设置过期时间，单位毫秒。等价于PSETEX命令。
- NX：指定的key不存在才会设置成功，用于添加指定的key。等价于SETNX命令。
- XX：指定的key必须存在才会设置成功，用于更新指定key的value。

l 说明：如果value字符串中带有空格，则该字符串需要使用双引号或单引号引起来，否则会认为set命令的参数数量不正确，报错。

setex与psetex

l 格式：SETEX/PSETEX key seconds value

l 功能：set expire，其不仅为key指定了value，还为其设置了生存时间。setex的单位为秒，psetex的单位为毫秒。

l 说明：如果 key 已经存在， 则覆写旧值。该命令类似于以下两个命令，不同之处是， SETEX 是一个原子性操作，关联值和设置生存时间两个动作会在同一时间内完成，该命令在 Redis 用作缓存时非常实用。

- SET key value
- EXPIRE key seconds # 设置生存时间

setnx

l 格式：SETNX key value

l 功能：SET if Not eXists，将 key 的值设为 value ，当且仅当 key 不存在。若给定的 key 已经存在，则 SETNX 不做任何动作。成功，返回1，否则，返回0。

l 说明：该命令等价于set key value nx

getset

l 格式：GETSET key value

l 功能：将给定 key 的值设为 value ，并返回 key 的旧值。

l 说明：当 key 存在但不是字符串类型时，返回一个错误；当 key 不存在时，返回 nil 。

mset与msetnx

l 格式：MSET/MSETNX key value [key value ...]

l 功能：同时设置一个或多个 key-value 对。

l 说明：如果某个给定 key 已经存在，那么 MSET会用新值覆盖原来的旧值，如果这不是你所希望的效果，请考虑使用 MSETNX命令：它只会在所有给定 key 都不存在的情况下进行设置操作。
MSET/MSETNX是一个原子性(atomic)操作，所有给定 key 都会在同一时间内被设置，某些给定 key 被更新而另一些给定 key 没有改变的情况不可能发生。该命令永不失败。

mget

l 格式：MGET key [key ...]

l 功能：返回所有(一个或多个)给定 key 的值。

l 说明：如果给定的 key 里面，有某个 key 不存在，那么这个 key 返回特殊值 nil 。因此，该命令永不失败。

append

l 格式：APPEND key value

l 功能：如果 key 已经存在并且是一个字符串， APPEND 命令将 value 追加到 key 原来的值的末尾。如果 key 不存在， APPEND 就简单地将给定 key 设为 value ，就像执行 SET key value 一样。

l 说明：追加 value 之后， key 中字符串的长度。

incr与decr

l 格式：INCR key 或 DECR key

l 功能：increment，自动递增。将 key 中存储的数字值增一。

decrement，自动递减。将key中存储的数字值减一。

l 说明：如果 key 不存在，那么 key 的值会先被初始化为0，然后再执行增一/减一操作。如果值不能表示为数字，那么返回一个错误提示。如果执行正确，则返回增一/减一后的值。

incrby与decrby

l 格式：INCRBY key increment 或 DECRBY key decrement

l 功能：将key中存储的数字值增加/减少指定的数值，这个数值只能是整数，可以是负数，但不能是小数。

l 说明：如果 key 不存在，那么 key 的值会先被初始化为0，然后再执行增/减操作。如果值不能表示为数字，那么返回一个错误提示。如果执行正确，则返回增/减后的值。

incrbyfloat

l 格式：INCRBYFLOAT key increment

l 功能：为 key 中所储存的值加上浮点数增量 increment 。

l 说明：与之前的说明相同。没有decrbyfloat命令，但increment为负数可以实现减操作效果。

strlen

l 格式：STRLEN key

l 功能：返回 key 所储存的字符串值的长度。

l 说明：当 key 储存的不是字符串值时，返回一个错误；当 key 不存在时，返回 0 。

getrange

l 格式：GETRANGE key start end

l 功能：返回 key 中字符串值的子字符串，字符串的截取范围由 start 和 end 两个偏移量决定，包括 start 和 end 在内。

l 说明：end必须要比start大。支持负数偏移量，表示从字符串最后开始计数，-1 表示最后一个字符，-2 表示倒数第二个，以此类推。

setrange

l 格式：SETRANGE key offset value

l 功能：用 value 参数替换给定 key 所储存的字符串值str，从偏移量 offset 开始。

l 说明：当offset值大于str长度时，中间使用零字节\x00填充，即0000 0000字节填充；对于不存在的 key 当作空串处理。

位操作命令

名称中包含BIT的命令，都是对二进制位的操作命令，例如，setbit、getbit、bitcount、bittop、bitfield，这些命令不常用。

4 Hash型Value操作命令

Redis存储数据的Value可以是一个Hash类型。Hash类型也称为Hash表、字典等。

Hash表就是一个映射表Map，也是由键-值对构成，为了与整体的key进行区分，这里的键称为field，值称为value。注意，Redis的Hash表中的field-value对均为String类型。

hset

l 格式：HSET key field value

l 功能：将哈希表 key 中的域 field 的值设为 value 。

l 说明：如果 key 不存在，一个新的哈希表被创建并进行 HSET 操作。如果域 field 已经存在于哈希表中，旧值将被覆盖。如果 field 是哈希表中的一个新建域，并且值设置成功，返回 1 。如果哈希表中域 field 已经存在且旧值已被新值覆盖，返回 0 。

1.1.2 hget

l 格式：HGET key field

l 功能：返回哈希表 key 中给定域 field 的值。

l 说明：当给定域不存在或是给定 key 不存在时，返回 nil 。

hmset

l 格式：HMSET key field value [field value ...]

l 功能：同时将多个 field-value (域-值)对设置到哈希表 key 中。

l 说明：此命令会覆盖哈希表中已存在的域。如果 key 不存在，一个空哈希表被创建并执行 HMSET 操作。如果命令执行成功，返回 OK 。当 key 不是哈希表(hash)类型时，返回一个错误。

hmget

l 格式：HMGET key field [field ...]

l 功能：按照给出顺序返回哈希表 key 中一个或多个域的值。

l 说明：如果给定的域不存在于哈希表，那么返回一个 nil 值。因为不存在的 key 被当作一个空哈希表来处理，所以对一个不存在的 key 进行 HMGET 操作将返回一个只带有 nil 值的表。

hgetall

l 格式：HGETALL key

l 功能：返回哈希表 key 中所有的域和值。

l 说明：在返回值里，紧跟每个域名(field name)之后是域的值(value)，所以返回值的长度是哈希表大小的两倍。若 key 不存在，返回空列表。若key中包含大量元素，则该命令可能会阻塞Redis服务。所以生产环境中一般不使用该命令，而使用hscan命令代替。

hsetnx

l 格式：HSETNX key field value

l 功能：将哈希表 key 中的域 field 的值设置为 value ，当且仅当域 field 不存在。

l 说明：若域 field 已经存在，该操作无效。如果 key 不存在，一个新哈希表被创建并执行 HSETNX 命令。

hdel

l 格式：HDEL key field [field ...]

l 功能：删除哈希表 key 中的一个或多个指定域，不存在的域将被忽略。

l 说明：返回被成功移除的域的数量，不包括被忽略的域。

hexits

l 格式：HEXISTS key field

l 功能：查看哈希表 key 中给定域 field 是否存在。

l 说明：如果哈希表含有给定域，返回 1 。如果不含有给定域，或 key 不存在，返回 0 。

hincrby与hincrbyfloat

l 格式：HINCRBY key field increment

l 功能：为哈希表 key 中的域 field 的值加上增量 increment 。hincrby命令只能增加整数值，而hincrbyfloat可以增加小数值。

l 说明：增量也可以为负数，相当于对给定域进行减法操作。如果 key 不存在，一个新的哈希表被创建并执行 HINCRBY 命令。如果域 field 不存在，那么在执行命令前，域的值被初始化为0。对一个储存字符串值的域 field 执行HINCRBY命令将造成一个错误。

hkeys与hvals

l 格式：HKEYS key 或 HVALS key

l 功能：返回哈希表 key 中的所有域/值。

l 说明：当 key 不存在时，返回一个空表。

hlen

l 格式：HLEN key

l 功能：返回哈希表 key 中域的数量。

l 说明：当 key 不存在时，返回 0 。

hstrlen

l 格式：HSTRLEN key field

l 功能：返回哈希表 key 中， 与给定域 field 相关联的值的字符串长度（string length）。

l 说明：如果给定的键或者域不存在， 那么命令返回 0 。

5 List型Value操作命令

Redis存储数据的Value可以是一个String列表类型数据。即该列表中的每个元素均为String类型数据。列表中的数据会按照插入顺序进行排序。不过，该列表的底层实际是一个无头节点的双向链表，所以对列表表头与表尾的操作性能较高，但对中间元素的插入与删除的操作的性能相对较差。

lpush/rpush

l 格式：LPUSH key value [value ...] 或 RPUSH key value [value ...]

l 功能：将一个或多个值 value 插入到列表 key 的表头/表尾（表头在左表尾在右）

l 说明：如果有多个 value 值，对于lpush来说，各个 value 会按从左到右的顺序依次插入到表头；对于rpush来说，各个 value 会按从左到右的顺序依次插入到表尾。如果 key 不存在，一个空列表会被创建并执行操作。当 key 存在但不是列表类型时，返回一个错误。执行成功时返回列表的长度。

llen

l 格式：LLEN key

l 功能：返回列表 key 的长度。

l 说明：如果 key 不存在，则 key 被解释为一个空列表，返回 0 。如果 key 不是列表类型，返回一个错误。

lindex

l 格式：LINDEX key index

l 功能：返回列表 key 中，下标为 index 的元素。列表从0开始计数。

l 说明：如果 index 参数的值不在列表的区间范围内(out of range)，返回 nil 。

lset

l 格式：LSET key index value

l 功能：将列表 key 下标为 index 的元素的值设置为 value 。

l 说明：当 index 参数超出范围，或对一个空列表（key 不存在）进行 LSET 时，返回一个错误。

lrange

l 格式：LRANGE key start stop

l 功能：返回列表 key 中指定区间[start, stop]内的元素，即包含两个端点。

l 说明：List的下标从0开始，即以 0 表示列表的第一个元素，以 1 表示列表的第二个元素，以此类推。也可以使用负数下标，以 -1 表示列表的最后一个元素， -2 表示列表的倒数第二个元素，以此类推。超出范围的下标值不会引起错误。如果 start 下标比列表的最大下标 还要大，那么 LRANGE返回一个空列表。如果 stop 下标比最大下标还要大，Redis将 stop 的值设置为最大下标。

lpushx与rpushx

l 格式：LPUSHX key value 或 **RPUSHX key value**

l 功能：将值 value 插入到列表 key 的表头/表尾，当且仅当 key 存在并且是一个列表。

l 说明：当 key 不存在时，命令什么也不做。若执行成功，则输出表的长度。

linsert

l 格式：LINSERT key BEFORE|AFTER pivot value

l 功能：将值 value 插入到列表 key 当中，位于元素 pivot 之前或之后。

l 说明：当 pivot 元素不存在于列表中时，不执行任何操作，返回-1；当 key 不存在时， key 被视为空列表，不执行任何操作，返回0；如果 key 不是列表类型，返回一个错误；如果命令执行成功，返回插入操作完成之后，列表的长度。

lpop / rpop

l 格式：LPOP key [count] 或 RPOP key [count]

l 功能：从列表key的表头/表尾移除count个元素，并返回移除的元素。count默认值1

l 说明：当 key 不存在时，返回 nil

blpop / brpop

l 格式：BLPOP key [key ...] timeout 或 BRPOP key [key ...] timeout

l 功能：BLPOP/BRPOP是列表的阻塞式(blocking)弹出命令。它们是 LPOP/RPOP 命令的阻塞版本，当给定列表内没有任何元素可供弹出的时候，连接将被 BLPOP/BRPOP命令阻塞，直到等待timeout超时或发现可弹出元素为止。当给定多个 key 参数时，按参数 key 的先后顺序依次检查各个列表，弹出第一个非空列表的头元素。timeout为阻塞时长，单位为秒，其值若为0，则表示只要没有可弹出元素，则一

直阻塞。

l 说明：假如在指定时间内没有任何元素被弹出，则返回一个 nil 和等待时长。反之，返回一个含有两个元素的列表，第一个元素是被弹出元素所属的 key ，第二个元素是被弹出元素的值。

rpoplpush

l 格式：RPOPLPUSH source destination

l 功能：命令 RPOPLPUSH 在一个原子时间内，执行以下两个动作：

- 将列表 source 中的最后一个元素(尾元素)弹出，并返回给客户端。
- 将 source 弹出的元素插入到列表 destination ，作为 destination 列表的的头元素。

如果 source 不存在，值 nil 被返回，并且不执行其他动作。如果 source 和 destination 相同，则列表中的表尾元素被移动到表头，并返回该元素，可以把这种特殊情况视作列表的旋转(rotation)操作。

brpoplpush

l 格式：BRPOPLPUSH source destination timeout

l 功能：BRPOPLPUSH 是 RPOPLPUSH 的阻塞版本，当给定列表 source 不为空时， BRPOPLPUSH 的表现和 RPOPLPUSH 一样。当列表 source 为空时， BRPOPLPUSH命令将阻塞连接，直到等待超时，或有另一个客户端对 source 执行 LPUSH或 RPUSH 命令为止。timeout为阻塞时长，单位为秒，其值若为 0，则表示只要没有可弹出元素，则一直阻塞。

l 说明：假如在指定时间内没有任何元素被弹出，则返回一个 nil 和等待时长。反之，返回一个含有两个元素的列表，第一个元素是被弹出元素的值，第二个元素是等待时长。

lrem

l 格式：LREM key count value

l 功能：根据参数 count 的值，移除列表中与参数 value 相等的元素。count 的值可以是以下几种：

- count > 0 : 从表头开始向表尾搜索，移除与 value 相等的元素，数量为 count 。
- count < 0 : 从表尾开始向表头搜索，移除与 value 相等的元素，数量为 count 的绝对值。
- count = 0 : 移除表中所有与 value 相等的值。

l 说明：返回被移除元素的数量。当 key 不存在时，LREM 命令返回 0 ，因为不存在的 key 被视作空表(empty list)。

ltrim

l 格式：LTRIM key start stop

l 功能：对一个列表进行修剪(trim)，就是说，让列表只保留指定区间内的元素，不在指定区间之内的元素都将被删除。

l 说明：下标(index)参数 start 和 stop 都以 0 为底，也就是说，以 0 表示列表的第一个元素，以 1 表示列表的第二个元素，以此类推。也可以使用负数下标，以 -1 表示列表的最后一个元素， -2 表示列表的倒数第二个元素，以此类推。当 key 不是列表类型时，返回一个错误。如果 start 下标比列表的最大下标 end (LLEN list 减去 1)还要大，或者 start > stop ， LTRIM 返回一个空列表，因为 LTRIM 已经将整个列表清空。如果 stop 下标比 end 下标还要大，Redis将 stop 的值设置为 end 。

6 Set型Value操作命令

Redis存储数据的Value可以是一个Set集合，且集合中的每一个元素均String类型。Set与List非常相似，但不同之处是Set中的元素具有无序性与不可重复性，而List则具有有序性与可重复性。

Redis中的Set集合与Java中的Set集合的实现相似，其底层都是value为null的hash表。也正因为此，才会引发无序性与不可重复性。

sadd

l 格式：SADD key member [member ...]

l 功能：将一个或多个 member 元素加入到集合 key 当中，已经存在于集合的 member 元素将被忽略。

l 说明：假如 key 不存在，则创建一个只包含 member 元素作成员的集合。当 key 不是集合类型时，返回一个错误。

smembers

l 格式：SMEMBERS key

l 功能：返回集合 key 中的所有成员。

l 说明：不存在的 key 被视为空集合。若key中包含大量元素，则该命令可能会阻塞Redis服务。所以生产环境中一般不使用该命令，而使用sscan命令代替。

scard

l 格式：SCARD key

l 功能：返回Set集合的长度

l 说明：当 key 不存在时，返回 0 。

sismember

l 格式：SISMEMBER key member

l 功能：判断 member 元素是否集合 key 的成员。

l 说明：如果 member 元素是集合的成员，返回 1 。如果 member 元素不是集合的成员，或 key 不存在，返回 0 。

smove

l 格式：SMOVE source destination member

l 功能：将 member 元素从 source 集合移动到 destination 集合。

l 说明：如果 source 集合不存在或不包含指定的 member 元素，则 SMOVE 命令不执行任何操作，仅返回 0 。否则， member 元素从 source 集合中被移除，并添加到 destination 集合中去，返回1。当 destination 集合已经包含 member 元素时， SMOVE 命令只是简单地将 source 集合中的 member 元素删除。当 source 或 destination 不是集合类型时，返回一个错误。

srem

l 格式：SREM key member [member ...]

l 功能：移除集合 key 中的一个或多个 member 元素，不存在的 member 元素会被忽略，且返回成功移除的元素个数。

l 说明：当 key 不是集合类型，返回一个错误。

srandmember

l 格式：SRANDMEMBER key [count]

l 功能：返回集合中的count个随机元素。count默认值为1。

l 说明：若count 为正数，且小于集合长度，那么返回一个包含 count 个元素的数组，数组中的元素各不相同。如果 count 大于等于集合长度，那么返回整个集合。如果 count 为负数，那么返回一个包含 count 绝对值个元素的数组，但数组中的元素可能会出现重复。

spop

l 格式：SPOP key [count]

l 功能：移除并返回集合中的count个随机元素。count必须为正数，且默认值为1。

l 说明：如果 count 大于等于集合长度，那么移除并返回整个集合。

sdiff / sdiffstore

l 格式：SDIFF key [key ...] 或 SDIFFSTORE destination key [key ...]

l 功能：返回第一个集合与其它集合之间的差集。差集，difference。

l 说明：这两个命令的不同之处在于，sdiffstore不仅能够显示差集，还能将差集存储到指定的集合destination中。如果 destination 集合已经存在，则将其覆盖。不存在的 key 被视为空集。

sinter / sinterstore

l 格式：SINTER key [key ...] 或 SINTERSTORE destination key [key ...]

l 功能：返回多个集合间的交集。交集，intersection。

l 说明：这两个命令的不同之处在于，sinterstore不仅能够显示交集，还能将交集存储到指定的集合destination中。如果 destination 集合已经存在，则将其覆盖。不存在的 key 被视为空集。

sunion / sunionstore

l 格式：SUNION key [key ...] 或 SUNIONSTORE destination key [key ...]

l 功能：返回多个集合间的并集。并集，union。

l 说明：这两个命令的不同之处在于，sunionstore不仅能够显示并集，还能将并集存储到指定的集合destination中。如果 destination 集合已经存在，则将其覆盖。不存在的 key 被视为空集。

7 有序Set型Value操作命令

Redis存储数据的Value可以是一个有序Set，这个有序Set中的每个元素均String类型。有序Set与Set的不同之处是，有序Set中的每一个元素都有一个分值score，Redis会根据score的值对集合进行由小到大的排序。其与Set集合要求相同，元素不能重复，但元素的score可以重复。由于该类型的所有命令均是字母z开头，所以该Set也称为ZSet。

zadd

l 格式：ZADD key score member [[score member] [score member] ...]

l 功能：将一个或多个 member 元素及其 score 值加入到有序集 key 中的适当位置。

l 说明：score 值可以是整数值或双精度浮点数。如果 key 不存在，则创建一个空的有序集并执行 ZADD 操作。当 key 存在但不是有序集类型时，返回一个错误。如果命令执行成功，则返回被成功添加的新成员的数量，不包括那些被更新的、已经存在的成员。若写入的member值已经存在，但score值不同，则新的score值将覆盖老score。

1.1.2 zrange与zrevrange

l 格式：ZRANGE key start stop [WITHSCORES] 或 ZREVRANGE key start stop [WITHSCORES]

l 功能：返回有序集 key 中，指定区间内的成员。zrange命令会按 score 值递增排序，zrevrange命令会按score递减排序。具有相同 score 值的成员按字典序/逆字典序排列。可以通过使用 WITHSCORES 选项，来让成员和它的 score 值一并返回。

l 说明：下标参数从0开始，即 0 表示有序集第一个成员，以 1 表示有序集第二个成员，以此类推。也可以使用负数下标，-1 表示最后一个成员，-2 表示倒数第二个成员，以此类推。超出范围的下标并不会引起错误。例如，当 start 的值比有序集的最大下标还要大，或是 start > stop 时， ZRANGE 命令只是简单地返回一个空列表。再比如 stop 参数的值比有序集的最大下标还要大，那么 Redis 将 stop 当作最大下标来处理。

若key中指定范围内包含大量元素，则该命令可能会阻塞Redis服务。所以生产环境中如果要查询有序集合中的所有元素，一般不使用该命令，而使用zscan命令代替。

1.1.3 zrangebyscore与zrevrangebyscore

l 格式：ZRANGEBYSCORE key min max [WITHSCORES][LIMIT offset count]

ZREVRANGEBYSCORE key max min [WITHSCORES][LIMIT offset count]

l 功能：返回有序集 key 中，所有 score 值介于 min 和 max 之间(包括等于 min 或 max)的成员。有序集成员按 score 值递增/递减次序排列。具有相同 score 值的成员按字典序/逆字典序排列。可选的 LIMIT 参数指定返回结果的数量及区间(就像SQL中的 SELECT LIMIT offset, count)，注意当 offset 很大时，定位 offset 的操作可能需要遍历整个有序集，此过程效率可能会较低。可选的 WITHSCORES 参数决定结果集是单单返回有序集的成员，还是将有序集成员及其 score 值一起返回。

l 说明：min 和 max的取值是正负无穷大的。默认情况下，区间的取值使用闭区间 (小于等于或大于等于)，也可以通过给参数前增加左括号“(”来使用可选的开区间 (小于或大于)。

zcard

l 格式：ZCARD key

l 功能：返回集合的长度

l 说明：当 key 不存在时，返回 0 。

zcount

l 格式：ZCOUNT key min max

l 功能：返回有序集 key 中， score 值在 min 和 max 之间(默认包括 score 值等于 min 或 max)的成员的数量。

zscore

l 格式：ZSCORE key member

l 功能：返回有序集 key 中，成员 member 的 score 值。

l 说明：如果 member 元素不是有序集 key 的成员，或 key 不存在，返回 nil 。

zincrby

l 格式：ZINCRBY key increment member

l 功能：为有序集 key 的成员 member 的 score 值加上增量 increment 。increment值可以是整数值或双精度浮点数。

l 说明：可以通过传递一个负数值 increment ，让 score 减去相应的值。当 key 不存在，或 member 不是 key 的成员时， ZINCRBY key increment member 等同于 ZADD key increment member 。当 key 不是有序集类型时，返回一个错误。命令执行成功，则返回member 成员的新 score 值。

zrank与zrevrank

l 格式：ZRANK key member 或 ZREVRANK key member

l 功能：返回有序集 key 中成员 member 的排名。zrank命令会按 score 值递增排序，zrevrank命令会按score递减排序。

l 说明：score 值最小的成员排名为 0 。如果 member 不是有序集 key 的成员，返回 nil 。

zrem

l 格式：ZREM key member [member ...]

l 功能：移除有序集 key 中的一个或多个成员，不存在的成员将被忽略。

l 说明：当 key 存在但不是有序集类型时，返回一个错误。执行成功，则返回被成功移除的成员的数量，不包括被忽略的成员。

zremrangebyrank

l 格式：ZREMRANGEBYRANK key start stop

l 功能：移除有序集 key 中，指定排名(rank)区间内的所有成员。

l 说明：排名区间分别以下标参数 start 和 stop 指出，包含 start 和 stop 在内。排名区间参数从0开始，即 0 表示排名第一的成员， 1 表示排名第二的成员，以此类推。也可以使用负数表示，-1 表示最后一个成员，-2 表示倒数第二个成员，以此类推。命令执行成功，则返回被移除成员的数量。

zremrangebyscore

l 格式：ZREMRANGEBYSCORE key min max

l 功能：移除有序集 key 中，所有 score 值介于 min 和 max 之间(包括等于 min 或 max)的成员。

l 说明：命令执行成功，则返回被移除成员的数量。

zrangebylex

l 格式：ZRANGEBYLEX key min max [LIMIT offset count]

l 功能：该命令仅适用于集合中所有成员都具有相同分值的情况。当有序集合的所有成员都具有相同的分值时，有序集合的元素会根据成员的字典序（lexicographical ordering）来进行排序。即这个命令返回给定集合中元素值介于 min 和 max 之间的成员。如果有序集合里面的成员带有不同的分值， 那么命令的执行结果与zrange key效果相同。

l 说明：合法的 min 和 max 参数必须包含左小括号“(”或左中括号“[”，其中左小括号“(”表示开区间，而左中括号“[”则表示闭区间。min或max也可使用特殊字符“+”和“-”，分别表示正无穷大与负无穷大。

zlexcount

l 格式：ZLEXCOUNT key min max

l 功能：该命令仅适用于集合中所有成员都具有相同分值的情况。该命令返回该集合中元素值本身（而非score值）介于 min 和 max 范围内的元素数量。

zremrangebylex

l 格式：ZREMRANGEBYLEX key min max

l 功能：该命令仅适用于集合中所有成员都具有相同分值的情况。该命令会移除该集合中元素值本身介于 min 和 max 范围内的所有元素。

8 BitMap操作命令

BitMap简介

BitMap是Redis 2.2.0版本中引入的一种新的数据类型。该数据类型本质上就是一个仅包含0和1的二进制字符串。而其所有相关命令都是对这个字符串二进制位的操作。用于描述该字符串的属性有三个：key、offset、bitValue。

l key：BitMap是Redis的key-value中的一种Value的数据类型，所以该Value一定有其对应的key。

l offset：每个BitMap数据都是一个字符串，字符串中的每个字符都有其对应的索引，该索引从0开始计数。该索引就称为每个字符在该BitMap中的偏移量offset。这个offset的值的范围是[0，232-1]，即该offset的最大值为4G-1，即4294967295，42亿多。

l bitValue：每个BitMap数据中都是一个仅包含0和1的二进制字符串，每个offset位上的字符就称为该位的值bitValue。bitValue的值非0即1。

setbit

l 格式：SETBIT key offset value

l 功能：为给定key的BitMap数据的offset位置设置值为value。其返回值为修改前该offset位置的bitValue

l 说明：对于原BitMap字符串中不存在的offset进行赋值，字符串会自动伸展以确保它可以将 value 保存在指定的offset上。当字符串值进行伸展时，空白位置以 0 填充。当然，设置的value只能是0或1。不过需要注意的是，对使用较大offset的SETBIT操作来说，内存分配过程可能造成Redis服务器被阻塞。

getbit

格式：GETBIT key offset

功能：对 key 所储存的BitMap字符串值，获取指定offset偏移量上的位值bitValue。

说明：当 offset 比字符串值的长度大，或者 key 不存在时，返回 0 。

bitcount

l 格式：BITCOUNT key [start] [end]

l 功能：统计给定字符串中被设置为 1 的bit位的数量。一般情况下，统计的范围是给定的整个BitMap字符串。但也可以通过指定额外的 start 或 end 参数，实现仅对指定字节范围内字符串进行统计，包括 start 和 end 在内。注意，这里的start与end的单位是字节，不是bit，并且从0开始计数。

l 说明：start 和 end 参数都可以使用负数值： -1 表示最后一个字节， -2 表示倒数第二个字节，以此类推。另外，对于不存在的 key 被当成是空字符串来处理，因此对一个不存在的 key 进行 BITCOUNT 操作，结果为 0 。

bitpos

l 格式：BITPOS key bit [start] [end]

l 功能：返回key指定的BitMap中第一个值为指定值 bit(非0即1) 的二进制位的位置。pos，即position，位置。在默认情况下， 命令将检测整个BitMap，但用户也可以通过可选的 start 参数和 end 参数指定要检测的范围。

l 说明：start与end的意义与bitcount命令中的相同。

bitop

l 格式：BITOP operation destkey key [key ...]

l 功能：对一个或多个BitMap字符串 key 进行二进制位操作，并将结果保存到 destkey 上。

l operation 可以是 AND 、 OR 、 NOT 、 XOR 这四种操作中的任意一种：

- BITOP AND destkey key [key ...]： 对一个或多个 BitMap执行按位与操作，并将结果保存到 destkey
- BITOP OR destkey key [key ...]： 对一个或多个BitMap执行按位或操作，并将结果保存到 destkey
- BITOP XOR destkey key [key ...]： 对一个或多个BitMap执行按位异或操作，并将结果保存到 destkey
- BITOP NOT destkey key： 对给定BitMap执行按位非操作，并将结果保存到 destkey

l 说明：

- 除了 NOT 操作之外，其他操作都可以接受一个或多个 BitMap 作为输入。
- 除了NOT操作外，其他对一个BitMap的操作其实就是一个复制。
- 如果参与运算的多个BitMap长度不同，较短的BitMap会以0作为补充位与较长BitMap运算，且运算结果长度与较长BitMap的相同。

9 HyperLogLog操作命令

HyperLogLog简介

HyperLogLog是Redis 2.8.9版本中引入的一种新的数据类型，其意义是hyperlog log，超级日志记录。该数据类型可以简单理解为一个set集合，集合元素为字符串。但实际上HyperLogLog是一种基数计数概率算法，通过该算法可以利用极小的内存完成独立总数的统计。其所有相关命令都是对这个“set集合”的操作。

HyperLogLog算法是由法国人Philippe Flajolet博士研究出来的，Redis的作者Antirez为了纪念Philippe Flajolet博士对组合数学和基数计算算法分析的研究，在设计HyperLogLog命令的时候使用了Philippe Flajolet姓名的英文首字母PF作为前缀。遗憾的是Philippe Flajolet博士于2011年3月22日因病在巴黎辞世。

pfadd

l 格式：PFADD key element [element ...]

l 功能：将任意数量的元素添加到指定的 HyperLogLog 集合里面。如果内部存储被修改了返回1，否则返回0。

pfcount

l 格式：PFCOUNT key [key ...]

l 功能：该命令作用于单个key时，返回给定key的HyperLogLog集合的近似基数；该命令作用于多个key时，返回所有给定key的HyperLogLog集合的并集的近似基数；如果key不存在，则返回0。

pfmerge

l 格式：PFMERGE destkey sourcekey [sourcekey ...]

l 功能：将多个HyperLogLog集合合并为一个HyperLogLog集合，并存储到destkey中，合并后的HyperLogLog的基数接近于所有sourcekey的HyperLogLog集合的并集。

10 Geospatial操作命令

Geospatial简介

Geospatial，地理空间。

Redis在3.2版本中引入了Geospatial这种新的数据类型。该类型本质上仍是一种集合，只不过集合元素比较特殊，是一种由三部分构成的数据结构，这种数据结构称为空间元素：

l 经度：longitude。有效经度为[-180，180]。正的表示东经，负的表示西经。

l 纬度：latitude。有效纬度为[-85.05112878， 85.05112878]。正的表示北纬，负的表示南纬。

l 位置名称：为该经纬度所标注的位置所命名的名称，也称为该Geospatial集合的空间元素名称。

通过该类型可以设置、查询某地理位置的经纬度，查询某范围内的空间元素，计算两空间元素间的距离等。

geoadd

l 格式：GEOADD key longitude latitude member [longitude latitude member ...]

l 功能：将一到多个空间元素添加到指定的空间集合中。

l 说明：当用户尝试输入一个超出范围的经度或者纬度时，该命令会返回一个错误。

geopos

l 格式：GEOPOS key member [member ...]

l 功能：从指定的地理空间中返回指定元素的位置，即经纬度。

l 说明：因为 该命令接受可变数量元素作为输入，所以即使用户只给定了一个元素，命令也会返回数组。

geodist

l 格式：GEODIST key member1 member2 [unit]

l 功能：返回两个给定位置之间的距离。其中unit必须是以下单位中的一种：

- m： 米，默认
- km： 千米

- mi： 英里
- ft：英尺

l 说明：如果两个位置之间的其中一个不存在， 那么命令返回空值。另外，在计算距离时会假设地球为完美的球形， 在极限情况下， 这一假设最大会造成 0.5% 的误差。

geohash

l 格式：GEOHASH key member [member ...]

l 功能：返回一个或多个位置元素的 Geohash值。

l 说明：GeoHash是一种地址编码方法。他能够把二维的空间经纬度数据编码成一个字符串。该值主要用于底层应用或者调试， 实际中的作用并不大。

georadius

l 格式：GEORADIUS key longitude latitude radius m|km|ft|mi [WITHCOORD] [WITHDIST] [WITHHASH] [ASC|DESC] [COUNT count]

l 功能：以给定的经纬度为中心，返回指定地理空间中包含的所有位置元素中，与中心距离不超过给定半径的元素。返回时还可携带额外的信息：

- WITHDIST： 在返回位置元素的同时，将位置元素与中心之间的距离也一并返回。 距离的单位和用户给定的范围单位保持一致。
- WITHCOORD： 将位置元素的经维度也一并返回。
- WITHHASH：将位置元素的Geohash也一并返回，不过这个hash以整数形式表示

命令默认返回未排序的位置元素。 通过以下两个参数，用户可以指定被返回位置元素的排序方式：

- ASC： 根据中心的位置，按照从近到远的方式返回位置元素。
- DESC： 根据中心的位置，按照从远到近的方式返回位置元素。

l 说明：在默认情况下， 该命令会返回所有匹配的位置元素。虽然用户可以使用 COUNT 选项去获取前N个匹配元素，但因为命令在内部可能会需要对所有被匹配的元素进行处理，所以在对一个非常大的区域进行搜索时，即使使用 COUNT 选项去获取少量元素，该命令的执行速度也可能会非常慢。

georadiusbymember

l 格式：GEORADIUSBYMEMBER key member radius m|km|ft|mi [WITHCOORD] [WITHDIST] [WITHHASH] [ASC|DESC] [COUNT count]

l 功能：这个命令和 GEORADIUS 命令一样，都可以找出位于指定范围内的元素，但该命令的中心点是由位置元素形式给定的，而不是像 GEORADIUS 那样，使用输入的经纬度来指定中心点。

l 说明：返回结果中也是包含中心点位置元素的

11 发布/订阅命令

消息系统

发布/订阅，即pub/sub，是一种消息通信模式：发布者也称为消息生产者，生产和发送消息到存储系统；订阅者也称为消息消费者，从存储系统接收和消费消息。这个存储系统可以是文件系统FS、消息中间件MQ、数据管理系统DBMS，也可以是Redis。整个消息发布者、订阅者与存储系统称为消息系统。

消息系统中的订阅者订阅了某类消息后，只要存储系统中存在该类消息，其就可不断的接收并消费这些消息。当存储系统中没有该消息后，订阅者的接收、消费阻塞。而当发布者将消息写入到存储系统后，会立即唤醒订阅者。当存储系统放满时，不同的发布者具有不同的处理方式：有的会阻塞发布者的发布，等待可用的存储空间；有的则会将多余的消息丢失。

当然，不同的消息系统消息的发布/订阅方式也是不同的。例如RocketMQ、Kafka等消息中间件构成的消息系统中，发布/订阅的消息都是以主题Topic分类的。而Redis构成的消息系统中，发布/订阅的消息都是以频道Channel分类的。

subscribe

l 格式：SUBSCRIBE channel [channel ...]

l 功能：Redis 客户端通过一个subscribe命令可以同时订阅任意数量的频道。在输出了订阅了主题后，命令处于阻塞状态，等待相关频道的消息。

psubscribe

l 格式：PSUBSCRIBE pattern [pattern ...]

l 功能：订阅一个或多个符合给定模式的频道。

l 说明：这里的模式只能使用通配符 。*例如， it* 可以匹配所有以 it 开头的频道，像 it.news、it.blog、it.tweets 等；news.*可以匹配所有以news.开头的频道，像news.global.today、 news.it等。

publish

l 格式：PUBLISH channel message

l 功能：Redis 客户端通过一条publish命令可以发布一个频道的消息。返回值为接收到该消息的订阅者数量。

unsubscribe

l 格式：UNSUBSCRIBE [channel [channel ...]]

l 功能：Redis客户端退订指定的频道。

l 说明：如果没有频道被指定，也就是一个无参数的UNSUBSCRIBE命令被执行，那么客户端使用SUBSCRIBE命令订阅的所有频道都会被退订。在这种情况下，命令会返回一个信息，告知客户端所有被退订的频道。

punsubscribe

l 格式：PUNSUBSCRIBE [pattern [pattern ...]]

l 功能：退订一个或多个符合给定模式的频道。

l 说明：这里的模式只能使用通配符 *。如果没有频道被指定，其效果与 SUBSCRIBE命令相同，客户端将退订所有订阅的频道。

pubsub

l 格式：PUBSUB [argument [argument ...]]

l 功能：PUBSUB 是一个查看订阅与发布系统状态的内省命令集，它由数个不同格式的子命令组成，下面分别介绍这些子命令的用法。

pubsub channels

l 格式：PUBSUB CHANNELS [pattern]

l 功能：列出当前所有的活跃频道。活跃频道指的是那些至少有一个订阅者的频道。

l 说明：pattern 参数是可选的。如果不给出 pattern 参数，将会列出订阅/发布系统中的所有活跃频道。如果给出 pattern 参数，那么只列出和给定模式 pattern 相匹配的那些活跃频道。pattern中只能使用通配符*。

pubsub numsub

l 格式：PUBSUB NUMSUB [channel-1 ... channel-N]

l 功能：返回给定频道的订阅者数量。不给定任何频道则返回一个空列表。

pubsub numpat

l 格式：PUBSUB NUMPAT

l 功能：查询当前Redis所有客户端订阅的所有频道模式的数量总和

12 Redis事务

Redis的事务的本质是一组命令的批处理。这组命令在执行过程中会被顺序地、一次性全部执行完毕，只要没有出现语法错误，这组命令在执行期间是不会被中断。

Redis事务特性

Redis的事务仅保证了数据的一致性，不具有像DBMS一样的ACID特性。

l 这组命令中的某些命令的执行失败不会影响其它命令的执行，不会引发回滚。即不具备原子性。

l 这组命令通过乐观锁机制实现了简单的隔离性。没有复杂的隔离级别。

l 这组命令的执行结果是被写入到内存的，是否持久取决于Redis的持久化策略，与事务无关。

Redis事务实现

Redis事务通过三个命令进行控制。

- muti：开启事务
- exec：执行事务
- discard：取消事务